J Netw Syst Manage (2011) 19:257–277
DOI 10.1007/s10922-010-9176-7

Trust Management and Admission Control for Host-Based Collaborative Intrusion Detection

Carol Fung · Jie Zhang · Issam Aib · Raouf Boutaba

Published online: 29 September 2010

Abstract The accuracy of detecting an intrusion within a network of intrusion detection systems (IDSes) depends on the efficiency of collaboration between member IDSes. The security itself within this network is an additional concern that needs to be addressed. In this paper, we present a trust-based framework for secure and effective collaboration within an intrusion detection network (IDN). In particular, we design a trust model that allows each IDS to evaluate the trustworthiness of other IDSes based on its personal experience. We also propose an admission control algorithm for the IDS to manage the acquaintances it approaches for advice about intrusions. We discuss the effectiveness of our approach in protecting the IDN against common attacks. Additionally, experimental results demonstrate that our system yields significant improvement in detecting intrusions. The trust model further improves the robustness of the collaborative system against malicious attacks. The experimental results also support that our admission control algorithm is effective and fair, and creates incentives for collaboration.

Keywords Security · Intrusion detection systems · Acquaintance management · Collaboration networks · Peer-to-peer networks · Insider attack · Robustness

C. Fung (✉) · I. Aib · R. Boutaba
David R. Cheriton School of Computer Science, University of Waterloo, 200 University west, Waterloo, ON, Canada
e-mail: j22fung@cs.uwaterloo.ca

I. Aib
e-mail: iaib@uwaterloo.ca

R. Boutaba
e-mail: rboutaba@uwaterloo.ca

J. Zhang
School of Computer Engineering, Nanyang Technological University, Block N4 #02c-110 Nanyang Avenue, Singapore 639798, Singapore
e-mail: zhangj@ntu.edu.sg

1 Introduction

Intrusions over the Internet are becoming more dynamic and sophisticated. Intrusion Detection Systems (IDSes) identify intrusions by comparing observable behavior against suspicious patterns. They can be network-based (NIDS) or host-based (HIDS). Traditional IDSes work in isolation and may be easily compromised by unknown or new threats. An Intrusion Detection Network (IDN) is a collaborative IDS network intended to overcome this weakness by having each member IDS benefit from the collective knowledge and experience shared by other member IDSes. This enhances the overall accuracy of intrusion assessment as well as the ability of detecting new intrusion types.

Intrusion types include worms, spamware, viruses, denial-of-service (DoS), malicious logins, etc. The potential damage of these intrusions can be significant if they are not detected promptly. An example is the Conflicker worm which infected more than 3 million Microsoft server systems during the year of 2008–2009, with the estimated economic loss of 9 billion dollars [1]. Evidence shows that attackers tend to use a large amount of compromised nodes to launch coordinated attacks to avoid being detected by single host intrusion detection systems [2]. IDS collaboration can be an effective way to throttle or stop the spread of such contagious attacks.

Centralized collaboration of IDSes relies on a central server to gather alerts and analyze them, such as DShield [3] and CRIM [4]. This technique suffers from the performance bottleneck problem. In addition, the central server is a single point of failure and may become the target of denial-of-service attacks. The distributed collaboration of IDSes, such as that in Indra [5] and NetShield [6], can avoid the bottleneck problem. However, in such collaborative environments, malicious IDSes can degrade the performance of others by sending out false evaluations about intrusions. To protect an IDS in a collaborative network from malicious attacks, it is important to evaluate the trustworthiness of participating IDSes, especially when they are Host-based (HIDSes). Duma et al. [7] propose a simple trust management model to identify dishonest insiders. However, their model is vulnerable to some attacks which aim at compromising the trust model itself (see Sects. 5, 6 for more details).

In this work, we develop a robust trust management model that is suitable for distributed HIDS collaboration. Our model allows each HIDS to evaluate the trustworthiness of others based on its own experience with them. We also propose a framework for efficient HIDS collaboration using a peer-to-peer network and an admission control algorithm for the HIDSes to select and manage their collaborators. Our framework provides identity verification for participating HIDSes and creates incentives for collaboration amongst nodes. Our admission control is also effective, fair and incentive.

We evaluate our system based on a simulated collaborative HIDS network. HIDSes are distributed and may have different expertise levels in detecting intrusions. A HIDS may also become malicious in case it has been compromised (or the HIDS owner deliberately makes it become malicious). We also simulate several potential threats. Our experimental results demonstrate that our system yields significant improvement in detecting intrusions and is robust to various attacks as

compared to that of [7]. Our admission control mechanism is also demonstrated to allow HIDSes to effectively and fairly manage their collaborators, and is able to create incentives for collaboration.

The rest of the paper is organized as follows. Section 2 presents the collaborative HIDS framework and collaboration management mechanisms. Section 3 formalizes our trust management model. Section 4 describes the admission control algorithm used for nodes to select and manage their collaborators. Section 5 addresses common attacks on HIDS collaboration networks, and discuss how our trust model is able to cope with these attacks. Section 6 presents the different simulation settings used for experiments and discusses the obtained results. Section 7 discusses related work. Finally, Sect. 8 summarizes our contributions and proposes directions for future work.

2 HIDS Collaboration Framework

The purpose of this framework is to connect individual HIDSes so that they can securely communicate and cooperate with each other to achieve better intrusion detection. Figure 1 illustrates the key components of our framework. Collaboration is ensured by trust-based cooperation and peer-to-peer communication. The trust management model allows a HIDS to evaluate the trustworthiness of its neighbors based on its own experience with them. The P2P component provides network organization, management and communication between the HIDSes. There are three processes for achieving collaboration among HIDSes, which are explained in the following sub-sections.

2.1 Network Join Process

In our framework, each HIDS connects to other HIDSes over a peer-to-peer network. Before joining the network, a HIDS node needs to register to a trusted digital certificate authority (Fig. 1) and get a public and private key pair which

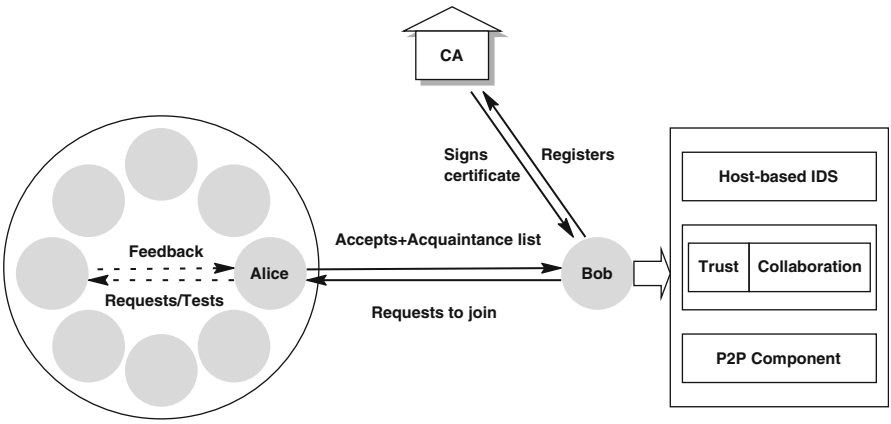

Fig. 1 HIDS collaboration framework

uniquely identifies it. Note that we identify the (IP-address, user) tuple. This is because a different IP-address means a different HIDS instance. In addition, a different user of the same machine may have a different configuration of its HIDS. However, only one user in a machine is activated at any time. After a node joins the HIDS network, it is provided with a preliminary *acquaintance list* by the initial connecting node (Alice in Fig. 1), which is referred by the bootstrap node or discovered by an automatic service discovery mechanism. This list is customizable and contains IDs (or public keys) of other nodes within the network along with their *trust* values. It serves as the contact list for collaboration. Note that the initial trust values for all new contacts are set to be low ($T_{stranger}$). The management of acquaintance lists by our admission control algorithm will be described in Sect. 4.

2.2 Test Messages

Each node sends out either *requests* for alert ranking (consultation), or *test messages*. A test message is a consultation request sent with the intention to evaluate the trustworthiness of another node in the acquaintance list. It is sent out in a way that makes it difficult to be distinguished from a real alert ranking request. The testing node knows the *severity* of the alert described in the test message and uses the received feedback to derive a trust value for the tested node. The IDMEF (Intrusion Detection Message Exchange Format) format can be used to exchange alert information between different IDSes. This technique helps in uncovering inexperienced and/or malicious nodes within the collaborative network.

2.3 Incentive Design

Our framework also provides *incentives* to motivate collaboration. Nodes that are asked for consultation will reply to only a number of requests in a certain period of time because of their limited bandwidth and computational resources. Thus, only highly trusted nodes will have higher priority of receiving help whenever needed. In this way, nodes are encouraged to build up their trust by collaborating with others. In addition, our system accepts the reply of "*don't know*" to requests in order to encourage active collaboration in the network. The details of this will in explained in Sect. 3.1.

3 Trust Management Model

This section describes the model we developed to establish trust relationships between HIDSes in the collaborative environment. We first describe how we evaluate the trustworthiness of a HIDS and then present a method to aggregate feedback responses from trusted IDSess.

3.1 Evaluating the Trustworthiness of a Node

The evaluation of the trustworthiness of a node is carried out using test messages generated periodically by a random Poisson process. After a node receives the

feedback for an alert evaluation, it assigns a satisfaction value to the feedback, which can be "very satisfied" (1.0), "satisfied" (0.5), "neutral" (0.3), "unsatisfied" (0.1), or "very unsatisfied" (0).

The trustworthiness of a node will be updated based on how satisfactory its feedbacks are. More specifically, the replies (feedbacks) from a node i are ordered from the most recent to the oldest according to the time t_k at which they have been received by node j. The *trustworthiness* of node i according to node j is then estimated as follows:

$$tw_i^j(n) = \frac{\sum_{k=0}^{n} S_k^{j,i} F^{t_k}}{\sum_{k=0}^{n} F^{t_k}} \tag{1}$$

where $S_k^{j,i} \in [0, 1]$ is the satisfaction of the reply k and n is the total number of replies. To deal with possible changes of the node's behavior over time, we use a *forgetting factor* F ($0 \leq F \leq 1$) which helps in assigning less weight to older feedback replies [8]. Compared to the work of Duma et al. [7], our model uses multiple satisfaction levels, and old experiences are forgotten exponentially. They only use two satisfaction levels (satisfied and unsatisfied), and all experiences in their model have the same impact.

We also allow a node to send a "don't know" answer to a request if it has no knowledge about the alert or is not confident with its ranking decision. However, some nodes may take advantage of this option by always providing "don't know" feedback responses so as to maintain their trust values. In order to encourage nodes to provide satisfactory feedback responses whenever possible, the trust value will be slowly updated every time when the node provides a "don't know" answer. The trustworthiness of a node i according to node j is then formulated as follows:

$$T_i^j = \left(tw_i^j - T_{stranger}\right)(1 - x)^m + T_{stranger}, \tag{2}$$

where x is the percentage of "don't know" answers from time t_0 to t_n. m is a positive incentive parameter (*forgetting sensitivity*) to control the severity of punishment to "don't know" replies, tw_i^j is the trust value without the integration of "don't know" answers (1), and $T_{stranger}$ is the default trust value of a stranger. As illustrated in Fig. 2, Eq. (2) causes trust values to converge to $T_{stranger}$ with the increase in the percentage of "don't know" answers. Eventually, the trust value will become that of a stranger. Equation 2 also allows the trust value of an untrusted node to slowly increase up to the level of a stranger by providing "don't know" answers. Therefore, nodes with little experience are motivated to provide "don't know" answers rather than to guess random alert rankings that are most likely incorrect.

3.2 Feedback Aggregation

Based on the history of trustworthiness, each node j requests alert consulting only from those nodes in its acquaintance list whose trust values are greater than a threshold th_t^j. We also consider *proximity*, which is a measure of the physical distance between the node that provides feedback and the node that sends the request. Physical location can be an important parameter in intrusion detection.

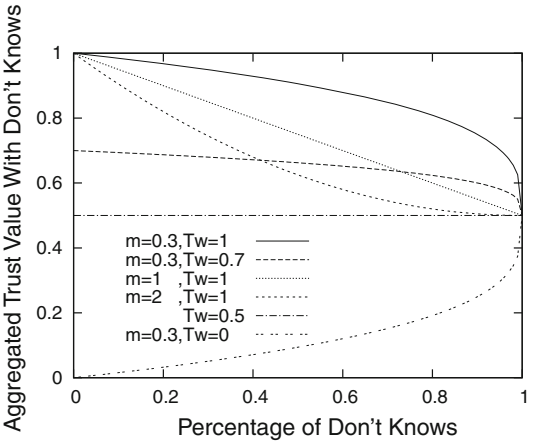

Fig. 2 Trust convergence curves with "don't know" answers

HIDSes that are located in the same or close by geographical region are likely to experience similar intrusions [9] and thus can help each other by broadcasting warnings of active threats. Feedback from nearby acquaintances is therefore more relevant than that from distant ones. We scale the proximity based on the region the node belongs to.

After receiving feedback from its acquaintances, node j aggregates the feedback using a *weighted majority* method as follows:

$$R_j(a) = \frac{\sum_{T_i^j \geq th_t^j} T_i^j D_i^j R_i(a)}{\sum_{T_i^j \geq th_t^j} T_i^j D_i^j}, \tag{3}$$

where $R_j(a)$ is the aggregated ranking of alert a from the feedback provided by each trust node belonging to the acquaintance list A_j of node j. $T_i^j (\in [0,1])$ is the trust value of node i according to node j. $D_i^j (\in [0,1])$ is the proximity weight of node i. th_t^j is the trust threshold set by node j. R_i (a) ($\in$ [0,1]) is the feedback ranking of alert a by node i. Our model only integrates feedback from trusted nodes while the model in [7] integrates feedback from all neighbors.

4 Admission Control

In this section, we propose an algorithm for a HIDS to manage its acquaintance list. In our proposed system, each HIDS (the host) in the IDN maintains a list of other HIDSes, called *acquaintances*. To learn the trust values of its acquaintances, a HIDS sends intrusion consultaions to the acquaintances and updates their trust values based on the satisfaction level of their feedback (see Sect. 3 for detailed computation). The intrusion consultations can be test messages or real alert evaluation requests. However, it is not recommended for a HIDS node to keep all the other nodes in its acquaintance list because maintaining a long acquaintance list requires much communication overhead and it does not scale. In this paper, we

propose that each HIDS maintains a fixed-length acquaintance list. Each node communicates with its acquaintances regularly to keep their trust values up to date. It also removes less trusted nodes and adds new nodes with higher trust values. Since it takes some time to learn the trust value of a new node, we propose that each HIDS also maintains a fixed-length *probation list*, where a new node needs to stay in probation for some period of time t_p before it can be moved to the acquaintance list. A node also communicates with the nodes in its probation list regularly to learn their trust values.

Suppose that node i has a list of n_i acquaintances. The corresponding trust values for those acquaintances are $\left\{T_1^i, T_2^i, \ldots, T_{n_i}^i\right\}$. We introduce the following parameters into the system. L_a is the maximum length of a HIDS's acquaintance list. L_p is the maximum length of a probation list. S_a and S_p denote the acquaintance list and the probation list respectively. t_u is the interval for updating the acquaintance and probation lists. t_p is the probation period.

The admission control algorithm is shown in Algorithm 1. The admission control algorithm starts by initializing S_a to emptySet and S_p to a number of L_p randomly selected nodes, (note that in a structured peer-to-peer system, a random node can be found by mapping a random value to a corresponding node through a distributed hash table). After initialization, an update process is triggered every t_u units of time. During the update process, each HIDS moves mature nodes which pass probation period from S_p into S_a, removes the acquaintance with the lowest trust value from S_a, and refills S_p with new random nodes. To avoid the cold start problem, the system fills S_a with nodes from S_p given the number of nodes in S_a is lower than the desired length L_a.

Several properties are desired for the admission control algorithm, which include convergence, fairness and incentive for collaboration. When the admission control algorithm is used, we expect to see that the acquaintance list of each node converges to a stable state. This is important since cooperation between HIDSes is long-term based. Frequently changing collaborators leads to resource waste because HIDSes need to learn the trustworthiness of new collaborators in probation periods. Fairness and incentive for collaboration are also important because they make the foundation for a long term healthy collaboration. We will evaluate our admission control algorithm based on these three metrics in Sect. 6.5.

5 Robustness Against Common Attacks

Trust management can effectively improve network collaboration and detect malicious HIDSes. However, the trust management itself may become the target of attacks and be compromised. In this section, we describe possible attacks and provide defense mechanisms against them.

Sybil attacks occur when a malicious node in the system creates a large amount of pseudonyms (fake identities) [10]. This malicious node uses fake identities to gain larger influence of the false alert ranking on others in the network. Our defense against sybil attacks relies on the design of an appropriate authentication

Algorithm 1 Manage acquaintance list and probation list of node i

Initialization:
$\mathbf{S_a} \Leftarrow \emptyset$
//fill $\mathbf{S_p}$ with randomly selected nodes
while $|\mathbf{S_p}| < L_p$ **do**
  $e \Leftarrow$ select random node
  $\mathbf{S_p} \Leftarrow \mathbf{S_p} \cup e$
end while
set new timer event(t_u, "**SpUpdate**")
at timer event ev of type "**SpUpdate**" **do**
//merge the expired nodes in the probation list into the acquaintance list
for each e in $\mathbf{S_p}$ **do**
  if $t(e) > t_p$//if node e passes probation period **then**
    $\mathbf{S_p} \Leftarrow \mathbf{S_p} \setminus e$
    $\mathbf{S_a} \Leftarrow \mathbf{S_a} \cup e$
  end if
end for
//remove the least trusted nodes from the acquaintance list till $|\mathbf{S}_a| \leq L_a$
while $|\mathbf{S_a}| > L_a$ **do**
  $a \Leftarrow$ least trusted node in $\mathbf{S_a}$
  $\mathbf{S_a} \Leftarrow \mathbf{S_a} \setminus a$
end while
//merge the most trusted nodes from $\mathbf{S_p}$ into $\mathbf{S_a}$ till $\mathbf{S_a}$ is filled
while $|\mathbf{S_a}| < L_a$ and $|\mathbf{S_a}| \neq 0$ **do**
  $a \Leftarrow$ most trusted node in $\mathbf{S_p}$
  $\mathbf{S_p} \Leftarrow \mathbf{S_p} \setminus p$
  $\mathbf{S_a} \Leftarrow \mathbf{S_a} \cup a$
end while
//refill $\mathbf{S_p}$ with randomly selected nodes
while $|\mathbf{S_p}| < L_p$ **do**
  $e \Leftarrow$ random node not in $\mathbf{S_a}$ or $\mathbf{S_p}$
  $\mathbf{S_p} \Leftarrow \mathbf{S_p} \cup e$
end while
set new timer event(t_u, "**SpUpdate**")
end do

mechanism. Authentication makes the registering of fake HIDes difficult. In our model, the certificate issuing authority (see Fig. 1) only allows one ID per IP address. In addition, our trust management model requires HIDSes to first build up their trust before they can affect the decisions of others, which is costly to do with many fake IDes. Thus, our security and trust mechanisms protect our collaborative network from sybil attacks.

Identity cloning attacks occur when a malicious node steals some node's identity and tries to communicate with others on its behalf. Our communication model is

based on asymmetric cryptography, where each node has a pair of public and private keys. The certificate authority certifies the ownership of key pairs and in this way protects the authenticity of node identities.

Newcomer attacks occur when a malicious node can easily register as a new user [11]. Such a malicious node creates a new ID for the purpose of erasing its bad history with other nodes in the network. Our model handles this type of attack by assigning low trust values to all newcomers, so their feedback on alerts is simply not considered by other nodes during the aggregation process.

Betrayal attacks occur when a trusted node suddenly turns into a malicious one and starts sending false alerts or even malware. A trust management system can be degraded dramatically because of this type of attacks. We employ a mechanism which is inspired by the social norm:

> It takes a long-time interaction and consistent good behavior to build up a high trust, while only a few bad actions to ruin it.

When a trustworthy node acts dishonestly, the forgetting factor (1) causes its trust value to drop down quickly, hence making it difficult for this node to deceive others or gain back its previous trust within a short time. The experimental result in Sect. 6.4 will show that our trust model is able to effectively cope with betrayal attacks.

Collusion attacks happen when a group of malicious nodes cooperate together by providing false alert rankings in order to compromise the network. In our system, nodes will not be adversely affected by collusion attacks. In our trust model each node relies on its own knowledge to detect dishonest nodes. In addition, we use test messages to uncover malicious nodes. Since the test messages are sent in a random manner, it will be difficult for malicious nodes to distinguish them from actual requests.

6 Simulations and Experimental Results

In this section, we present the experiments for evaluating the effectiveness and robustness of our trust-based HIDS collaboration framework.

6.1 Simulation Setting

In our simulation model, we have n nodes in the collaboration network randomly distributed in an $s \times s$ grid region. The proximity weight of nodes is anti-proportional to the distance between the nodes in the number of grid steps. The expertise levels of the nodes can be low (0.1), medium (0.5) or high (0.95). In the beginning, each node is assigned an initial acquaintance list based on the cost (proximity) of communication with other nodes. The initial trust values of all nodes in the acquaintance list are set to the stranger trust value ($T_{stranger}$). To test the trustworthiness of all nodes in the list, each node sends out test messages following a Poisson process with an average arrival rate λ_t. The intrusion detection expertise of a HIDS is modeled using a Beta function. An honest HIDS always generates feedback based on its truthful judgment, while a dishonest HIDS always sends

feedback opposite to its truthful judgment. The parameters used in the simulation are shown in Table 1.

To reflect a HIDS expertise level, we use a Beta distribution for the decision model of HIDSes. The Beta density function is expressed in (4):

$$f(p|\alpha,\beta) = \frac{1}{B(\alpha,\beta)} p^{\alpha-1}(1-p)^{\beta-1}, \tag{4}$$

$$B(\alpha,\beta) = \int_0^1 t^{\alpha-1}(1-t)^{\beta-1}dt, \tag{5}$$

We define α and β as:

$$\alpha = 1 + \frac{(1-D)}{D}\sqrt{\frac{E(b-1)}{1-E}}, \quad \beta = 1 + \frac{(1-D)(1-E)}{DE}\sqrt{\frac{E(b-1)}{1-E}} \tag{6}$$

where E is the expected ranking of the alert. D ($\in$[0,1]) denotes the difficulty level of an alert. Higher values for D are associated to attacks that are difficult to detect, i.e. many HIDSes fail to identify them. L ($\in$[0,1]) denotes the expertise level of a HIDS. A higher value for L reflects a higher probability of producing correct rankings for alerts. $f(p; \alpha, \beta)$ is the probability that the node with expertise level L answers with a value of p ($1 \geq p \geq 0$) to an alert of difficulty level D, $\alpha \geq 1$, $\beta \geq 1$, and $b = 1/(1-L)^2$. For a fixed difficulty level, this model assigns higher probabilities of producing correct ranking to nodes with higher levels of expertise. For a node with a fixed expertise level, it has a lower probability of producing correct rankings for alerts with higher D values. A node with expertise level 1 or an alert with difficulty level 0 represents the extreme case where the node can rank the alert accurately with guarantee. This is reflected in the Beta distribution by parameters $\alpha = \infty$ and $\beta = \infty$. A node with expertise level 0 or an alert with difficulty level 1 represents the extreme case where the node ranks the alert by picking up an answer randomly. This is reflected in the Beta distribution by parameters $\alpha = 1$ and $\beta = 1$ (Uniform distribution). Figure 3 shows the feedback probability distribution for HIDSes with

Table 1 Simulation parameters for experiments

Parameter	Value	Description
λ_t	5/day	Test messages frequency
F	0.9	Forgetting factor
$T_{stranger}$	0.5	Stranger trust value
th_t	0.8	Trust threshold
m	0.3	Forgetting sensitivity
s	4	Size of grid region
n	30 (+10)	Number of HIDS
L_a	6	Maximum acquaintance list length
L_p	2	Maximum probation list length
t_p	30	Probation period

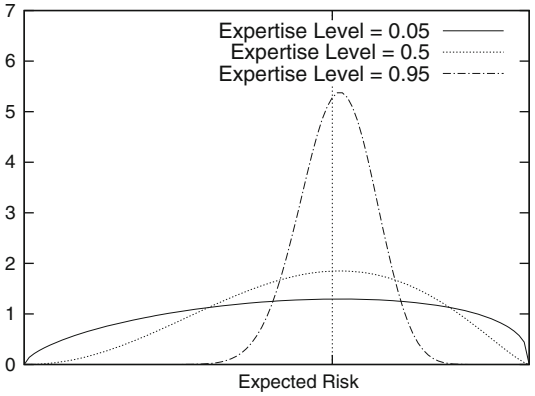

Fig. 3 Decision distribution for expertise levels

different expertise levels, where the expected risk level is fixed to 0.6 and the difficulty level of test messages is 0.5. From Fig. 3, we can see that the points on the curve for a higher expertise level (i.e. 0.95) lie closer to the correct risk level 0.6 on the X axis. This implies that if the expertise level of a node is higher, the node will have a higher probability of providing an answer that is close to the correct risk level.

6.2 Results for an Honest Environment

The first experiment studies the effectiveness of HIDS collaboration and the importance of trust management. In this experiment, all HIDSes are honest. 30 HIDSes are divided into three equally-sized groups, with expertise levels of 0.1, 0.5 and 0.95 respectively. We simulate the first 100 days to observe the trust values of nodes from each group, where nodes send only test messages to the others. Figure 4 shows the average trust values of nodes with different expertise levels. We can see that after 40–50 days, the trust values of all nodes converge to stable values. The nodes with higher expertise levels are able to gain higher trust. We also notice that the choice of parameters such as forgetting factor F and test message rate λ_t

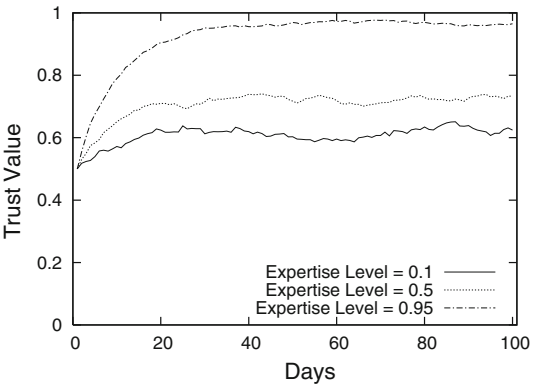

Fig. 4 Convergence of trust values for different expertise levels during the learning period

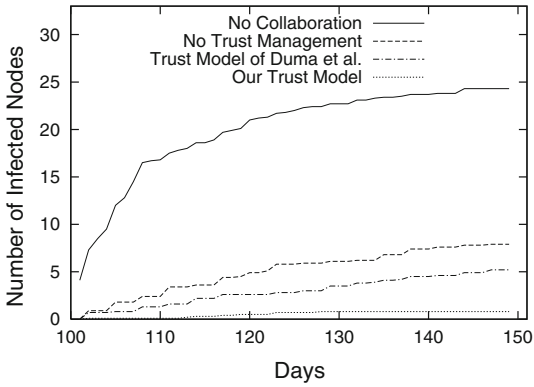

Fig. 5 Number of Infected nodes for an honest environment during the attack period

influences the learning speed. Lower F or higher λ_t leads to faster learning speed and thus the trust value takes less days to converge. However, low F brings high oscillation to the learned trust values and high λ_t causes high communication overhead. Our experiments are based on the moderate values $F = 0.9$ and $\lambda_t =$ 5/day.

Starting from day 101, we inject one random attack to all the nodes in the network in each day. The risk levels of the attacks are uniformly generated from [low, medium, high]. The difficulty levels of attacks are fixed to 0.5. Each HIDS in the network ranks the alert generated by the attack. If a node ranks "no risk" for any attack or ranks "low risk" for a high-risk attack, then it is assumed to be infected after the attack. We observe the total number of infected nodes in the network from day 101 to day 150 under different collaboration modes: no collaboration, collaboration without trust management, the trust model of Duma et al. [7] and our trust management method. The results of the total number of infected nodes are shown in Fig. 5. In this figure, we can see that the network in collaboration mode is more resistant to attacks than the network in the mode of no collaboration. Trust management models further improve the effectiveness of the collaboration. Our trust model performs better than that of Duma et al. [7]. Almost all attacks are detected and almost no node is infected after 50 days.

6.3 Results for an Environment With Dishonest Nodes

The purpose of the second experiment is to study the effectiveness of the collaboration model in a hazard situation where some nodes in the network are dishonest. We look at a case where only some expert nodes are dishonest because malicious expert nodes have the largest impact on the system.

In this experiment, we have 10 expert but dishonest nodes. There are two cases, without and with "don't know" replies. In the latter case, the percentages of "don't know" answers from nodes with expertise levels of 0.95, 0.5 and 0.1 are 0, 4 and 45% respectively. The forgetting sensitiveness parameter (m) varies from 0.3 to 2. Figure 6 shows the converged trust values of nodes with different expertise/honesty levels in "report all" case and "report don't know" case after 100 simulation days.

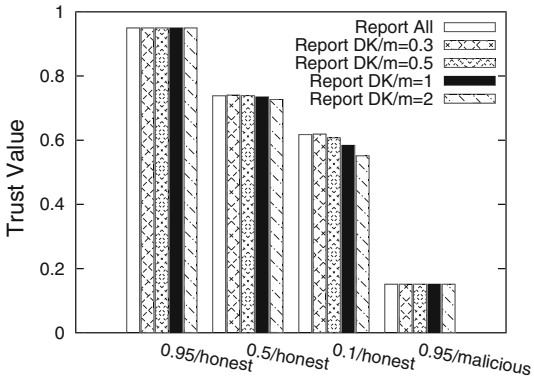

Fig. 6 Converged trust values for different expertise/honesty levels

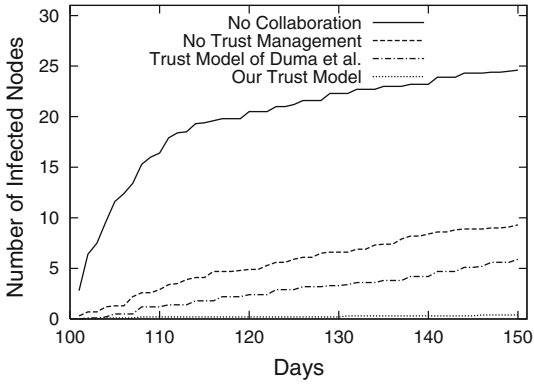

Fig. 7 Number of infected nodes for dishonest environment

When m is large, the punishment of reporting "don't know"'s becomes heavier. For $m = 0.3$, the nodes with medium (0.5) and low (0.1) expertise levels will be slightly rewarded. Therefore, we suggest using $m = 0.3$ to encourage non-expert nodes to report "don't know" when they are not confident about their replies.

After 100 days, we start injecting randomly generated attacks to all the nodes in the network at a rate of one attack per day. Figure 7 shows the total number of infected nodes under different collaboration modes. The number of infected nodes in the case of no collaboration is about the same as that in Fig. 5 because the HIDSes make decisions independently. When there is no trust model or using the model of Duma et al. [7] to detect malicious nodes, the total number of infected nodes is larger than the corresponding case in Fig. 5. The IDN hence suffers from malicious nodes. The number of infected nodes remains very small when using our trust model. This shows how important effective trust management is for a HIDS collaboration system.

6.4 Robustness of the Trust Model

The goal of this experiment is to study the robustness of our trust model against attacks. For the newcomer attack, malicious nodes white-wash their bad history and

re-register as new users to the system. However, it is quite difficult for a newcomer to succeed in our system. This is because it takes a long time for a newcomer to gain trust over the trust threshold. In our experiment, it takes about 15 days for an expert node to gain the trust of 0.8 to pass the threshold (as shown in Fig. 4).

The second possible threat is the betrayal attack, where a malicious node gains a high trust value and then suddenly starts to act dishonestly. This scenario happens, for example, when a node is compromised. To demonstrate the robustness of our model against this attack type, we add 10 expert nodes which spread opposite alert rankings on day 110. Figures 8 and 9 show the trust values of the betraying nodes and the number of correct attack rankings before and after the betrayal attack when using the trust model of Duma et al. and our trust model respectively. When using our trust model, we notice that the impact of the betrayal attack is smaller and the trust of malicious nodes drops down faster. This is because our trust model uses a forgetting factor. The trust values of dishonest nodes rely more on recent experience and therefore decrease faster. Our recovery phase is also shorter because we use a threshold to eliminate the impact of dishonest nodes. Once the trust values of malicious nodes drops below the trust threshold of 0.8, they are ignored in the alert consultation process and their impact is completely eliminated.

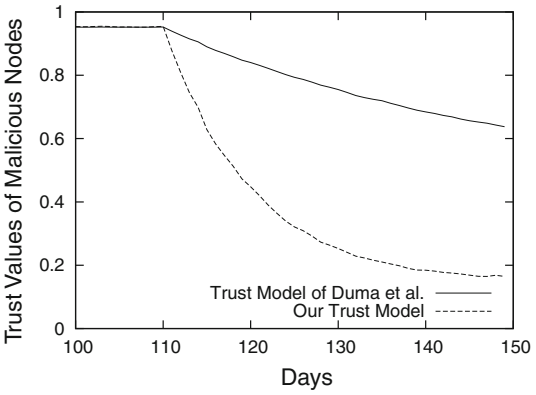

Fig. 8 Trust of malicious nodes

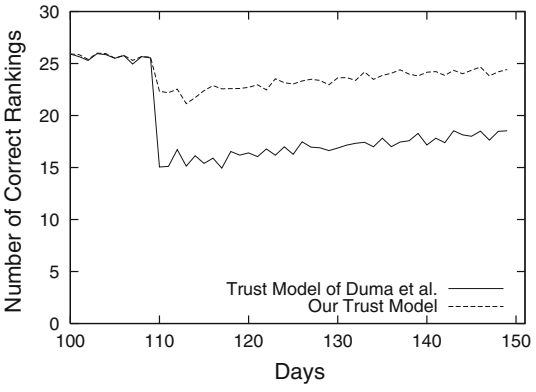

Fig. 9 The Impact of betrayal attack

6.5 Properties of the Admission Control

6.5.1 Convergence

In this experiment, we study the convergence of acquaintance list using our proposed admission control. Firstly, we simulate a network of 30 HIDSes divided into 10 equally-sized groups with expertise levels of 0.1, 0.2, …, 1.0, respectively. We set the maximum acquaintance list length $L_a = 6$ and the maximum probation list length $L_p = 2$. The updating interval is 5 days and the probation period is 30 days. Figure 10 plots the average expertise levels of the acquaintances for nodes with different expertise levels. We observe that in the first three rounds, the average expertise levels of acquaintances for all nodes converge to 0.55. After that, they diverge and converge to stable values depending on the expertise levels of the host nodes. This shows the convergence of our admission control algorithm.

We also count the number N of nodes which included a given node into their acquaintance lists. Figure 11 shows the average N for each group of nodes with similar expertise levels. We can see that nodes with higher expertise levels have a larger number of collaborators while nodes with low expertise levels have a smaller number of collaborators.. This is because the nodes with higher expertise levels are more likely to stay in the acquaintance list of others nodes while the nodes with lower expertise levels are easily replaced by other nodes with higher expertise levels.

6.5.2 Fairness

In the next experiment, we create a random IDN containing 100 nodes with random expertise levels uniformly distributed in [0,1]. All nodes are honest. We observe the converged average expertise level of nodes in the acquaintance list for all nodes in the network. The result is shown in Fig. 12. Notice how nodes with higher expertise levels end up having collaborators whose expertise levels are close by. This reflects a good fairness in the collaboration.

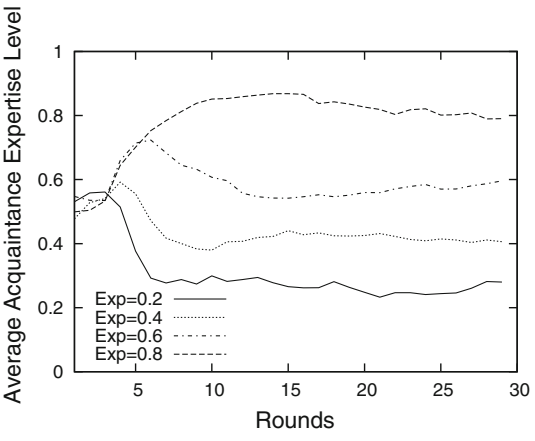

Fig. 10 Convergence of the acquaintance list

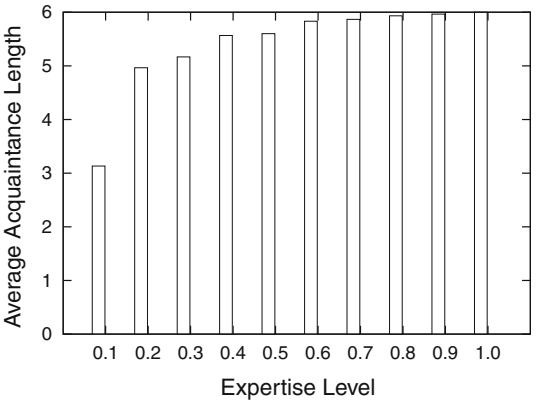

Fig. 11 The average acquaintance length

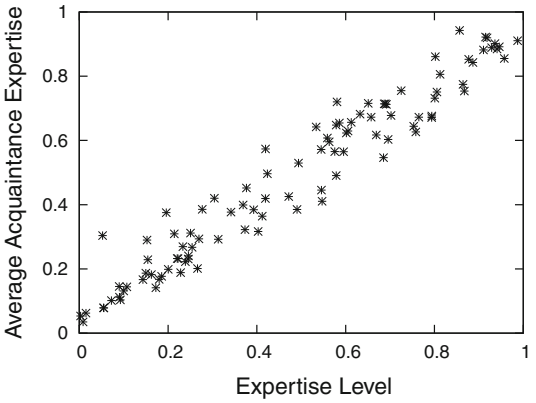

Fig. 12 Converged acquaintances average expertise levels for a random HIDN

6.5.3 Incentive for Collaboration

Next, we added another 50 nodes which are dishonest and have random expertise levels uniformly distributed between [0,1]. We observe the distribution of the acquaintances for all nodes after convergence. Figure 13 shows the distribution of acquaintances' expertise levels for all nodes in the network, where a negative expertise represents a dishonest node. Notice that honest nodes end up having few dishonest nodes in their acquaintance lists, while dishonest nodes tend to collaborate with other dishonest nodes. This reflects the balanced collaboration inventive induced by our admission control mechanism. Nodes are encouraged to behave honestly no matter what their expertise levels are. A dishonest node ends up not receiving help from honest ones.

7 Related Work

Most of the existing work on distributed collaborative intrusion detection relies on the assumption that all HIDSes are trusted and faithfully report intrusion events.

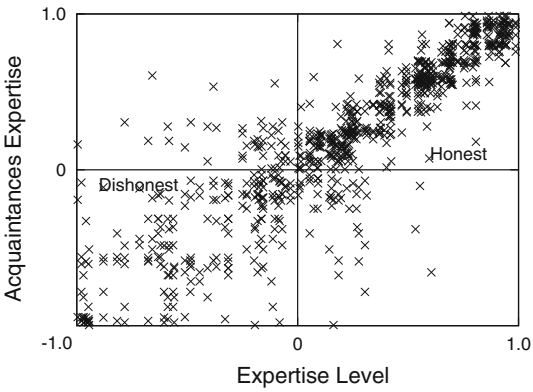

Fig. 13 Acquaintances expertise levels for a random HIDN with dishonest nodes

This is the case, for instance, for "Indra" [5] and the distributed intrusion alert fusion system called Cyber Disease Distributed Hash Table (CDDHT) proposed by Li et al. [12]. "Indra" distributes among peers in a P2P network information about attack attempts on different machines. In this way, the system can react proactively and the chance of noticing an attack increases. The "Indra" system also allows neighbors of peers to watch out for intrusion attempts in order to enhance security. The CDDHT system provides several load balancing schemes to evenly distribute intrusion alarms among the sensor fusion centers, in order to increase the scalability, fault-tolerance and robustness of the system. However, these systems are vulnerable to malicious IDSes. False information about intrusion events sent by malicious IDSes may heavily degrade the performance of these collaborative IDNs.

To protect an IDN, it is important to evaluate the trustworthiness of participating IDSes. ABDIAS [13] is a community based collaborative IDN where IDSes are organized into groups and exchange intrusion information to gain better intrusion detection accuracy. A simple majority-based voting system was proposed to detect compromised nodes. However, this voting-based system is vulnerable to colluded voting. Duma et al. [7] propose to address possibly malicious IDSes (peers) by introducing a trust-aware collaboration engine for correlating intrusion alerts. Their trust management scheme uses each peer's past experience to predict others' trustworthiness. However, their trust model is simplistic and does not address security issues within the collaborative network. For instance, in their system, the peer's past experience has the same impact on the final trust values of others, and therefore is vulnerable to betrayal attacks where compromised peers suddenly change their behavior. In our model, we use a forgetting factor when calculating trust, in order to rely more on the peer's recent experience and be suitable in a highly dynamic context of intrusion detection. A Dirichlet-based trust model proposed in [14] adopts a Bayesian model to calculate trust values and estimate the likely future behavior of an IDS based on its past history. The Bayesian model provides a theoretical foundation for the estimation. This model of also focuses on the scalability properties of the IDS collaborative network. It adopts a dynamic message rate mechanism based on the confidence of the trust estimation, so as to allow a peer to send fewer requests to some peers and more requests to others.

This work develops a detailed admission control model for IDSes to manage their acquaintance lists based on the trustworthiness of other IDSes in the network. We also incorporate the admission control model into our larger-scale simulation framework to fully testify the effectiveness of our trust management. The experimentation includes the results of the convergence of peers' acquaintance lists and the number of acquaintances for peers with different expertise levels. The design of our admission control algorithm and our particular proposal of allowing peers to send the "don't know" reply to requests provide incentives to motivate collaboration.

People in multi-agent systems have also been developing trust models to evaluate the trustworthiness of buying and selling agents for the application of e-marketplaces [8]. One of the earliest trust models developed by Marsh [15] computes the trustworthiness of selling agents by taking into account direct interactions between buying and selling agents. The trust-oriented learning strategy proposed by Tran and Cohen [16] uses reinforcement learning to determine the trustworthiness of selling agents, after the true value of delivered goods is evaluated and compared to the buying agent's expected value for the goods. Selling agents can be classified as untrustworthy if their trust values fall below a certain threshold and buying agents try to select the trustworthy selling agent with the highest expected value for the goods. The Beta Reputation System (BRS) of Whitby et al. [17] and the TRAVOS model of Teacy et al. [18] estimate the trustworthiness of a selling agent by employing a Beta probability density function representing a probability distribution of a continuous variable. Our model is distinguished from these trust models in several aspects. We introduce the concepts of expertise level and physical location in order to improve the accuracy of intrusion detection. We also allow IDSes to send test messages to better establish the trust relationships with others. The alert risk ranking is categorized into multiple levels as well. The REGRET model of Sabater et al. [19] offers a multi-dimensional view of trust that includes a social dimension taking into consideration the social relationships among agents. However, it is difficult to clearly indicate social relationships among IDSes in collaborative IDNs.

Different reputation models are proposed in distributed systems [20–23]. These reputation models allow peers to ask advice about other peers' trustworthiness when evaluating the trustworthiness of these other peers. For example, EigenTrust [20] uses iterative computation to calculate the reputation value of each node in the network based on the ratings of other nodes in the network. Jiang and Baras [21] uses a global reputation management to evaluate distributed trust by aggregating votes from all peers in the network. Sun et al. [22] propose for communication in distributed networks an entropy-based model and a probability-based model, which are used to calculate indirect trust, propagation trust and multi-path trust. These models involve a lot of overhead. Scalability becomes an important concern when employing these reputation models in collaborative IDNs. Another big concern is that IDSes can be easily compromised and become deceptive when reporting the trustworthiness of other IDSes. Numerous reputation models are proposed for peer-to-peer networks [24], such as PowerTrust [25], TrustGuard [26], and Fine-Grained reputation [27] are able to detect malicious nodes in the network. However, their main purpose is to detect deceiving nodes in P2P network and cannot be directly

adapted into intrusion detection networks to improve the intrusion detection accuracy. The BRS system [17], the TRAVOS model [18] and the work of Zhang and Cohen [8] focus on coping with inaccurate reputation information about selling agents shared by malicious buying agents in e-marketplaces. The result of their work may be adopted for the environment of collaborative IDNs. The third concern is that the reputation models may suffer from collusion attacks where a group of malicious IDSes cooperate together by providing false reputation information about some IDSes to, for example bad-mouth these targets. Incorporating reputation models into our trust management by addressing the above issues is left for future work.

8 Conclusions and Future Work

In this paper, we presented a trust-based HIDS collaboration framework that enhances intrusion detection within a host-based IDN and an admission control algorithm for participating HIDSes to select and manage collaborating HIDSes. The framework creates incentives for collaboration and we prove it is robust against common attacks in the collaborative network. The admission control algorithm adds fairness to the collaboration and encourages nodes to behave honestly in order to receive proper help from higher caliber HIDSes when need arises. The conducted simulations demonstrate the improved performance of our framework in detecting intrusions as well as its robustness against malicious attacks.

As future work, we will investigate the design of a communication protocol for the collaborative network and develop and deploy a real IDN using existing intrusion detection systems, which takes privacy and efficiency issues into consideration. We will also design a function for automatically measuring risk and difficulty levels of test messages, as well as feedback from HIDSes as input and automatically generates satisfaction levels of the received feedback as output.

Furthermore, we intend to extend our trust model to go beyond a generalized trust value for a HIDS. More specifically, since in practice HIDSes might have different expertise in detecting different types of intrusions, we would want to model the trustworthiness of a HIDS with respect to each individual type of intrusion. This will result in more effective trust management for assisting HIDSes to seek advice from truly helpful others. The subjectivity of HIDSes needs to be addressed when modeling their trustworthiness. HIDSes may have different subjective opinions on the risk levels of alerts. They can be more or less sensitive to certain intrusions.

Incentive design is another possible extension of our work. In the current protocol, the system may suffer from the free-riders problem where some nodes forward test messages to their neighbors, receive the appropriate rankings, and then forward the aggregated feedback back to the tester. Free-riders can create unnecessary traffic in the network and degrade the efficiency of the system. Honest nodes may be taken advantage of and deceived by dishonest “middle-agents”. In our future work, we will investigate this problem and enhance the incentive design to discourage free-riders and reward honest participants.

Finally, we also intend to evaluate the resistance of our framework against collusion attacks; as well as investigate its scalability in terms of the number of HIDSes, message rate and attack type. For example, a number of malicious nodes may collaboratively send a huge amount of test messages or real requests to block valid network transmission. To prevent this kind of excessive "test messages" attacks, each node can negotiate with all neighbors about the rate of the test messages sent to each other. A resource management protocol may also be needed to decide the test message rate among each pair of nodes.

Acknowledgments This work was supported in part by the Natural Science and Engineering Council of Canada (NSERC) Strategic program and in part by the WCU (World Class University) program through the Korea National Research Foundation funded by the Ministry of Education, Science and Technology (Project No. R31-2008-000-10100-0).

References

1. Danchev, D.: Conficker's estimated economic cost? $9.1 billion. http://www.zdnet.com/blog/security/confickers-estimated-economic-cost-91-billion/3207 (2009). Accessed 3 Aug 2010
2. Zhou, C., Leckie, C., Karunasekera, S.: A survey of coordinated attacks and collaborative intrusion detection. Comput. Secur. **29**(1), 124–140 (2010)
3. Ullrich, J.: DShield. http://www.dshield.org (2000). Accessed 3 Aug 2010
4. Cuppens, F., Miege, A.: Alert correlation in a cooperative intrusion detection framework. In: Proceedings of 2002 IEEE Symposium on Security and Privacy, pp. 202–215. (2002)
5. Janakiraman, R., Zhang, M.: Indra: a peer-to-peer approach to network intrusion detection and prevention. In: WET ICE 2003. Proceedings of the 12th IEEE International Workshops on Enabling Technologies. (2003)
6. Cai, M., Hwang, K., Kwok, Y., Song, S., Chen, Y.: Collaborative internet worm containment. IEEE Secur. Priv. **3**(3), 25–33 (2005)
7. Duma, C., Karresand, M., Shahmehri, N., Caronni, G.: A trust-aware, p2p-based overlay for intrusion detection. In: DEXA Workshops (2006)
8. Zhang, J., Cohen, R.: Trusting advice from other buyers in e-marketplaces: the problem of unfair ratings. In: ICEC '06, pp. 225–234. ACM, New York, NY (2006)
9. Aycock, J.: Painting the internet: a different kind of warhol worm. Technical report, TR2006-834-27. University of Calgary (2006)
10. Douceur, J.: The sybil attack. In: Peer-to-Peer Systems: First International Workshop, IPTPS 2002, Cambridge, MA, USA, 7–8 March 2002
11. Resnick, P., Kuwabara, K., Zeckhauser, R., Friedman, E.: Reputation systems. Commun. ACM **43**(12), 45–48 (2000)
12. Li, Z., Chen, Y., Beach, A.: Towards scalable and robust distributed intrusion alert fusion with good load balancing. In: LSAD '06 (2006)
13. Ghosh, A., Sen, S.: Agent-based distributed intrusion alert system. In: Proceedings of the 6th International Workshop on Distributed Computing (IWDCG04). Springer (2004)
14. Fung, C., Zhang, J., Aib, I., Boutaba, R.: Robust and scalable trust management for collaborative intrusion detection. In: Proceedings of the Eleventh IFIP/IEEE International Symposium on Integrated Network Management (IM) (2009)
15. Marsh, S.: Formalising Trust as a Computational Concept. Ph.D. thesis, Department of Mathematics and Computer Science, University of Stirling (1994)
16. Tran, T., Cohen, R.: Improving user satisfaction in agent-based electronic marketplaces by reputation modeling and adjustable product quality. In: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS), pp. 828–835. (2004)
17. Whitby, A., Jøsang, A., Indulska, J.: Filtering out unfair ratings in bayesian reputation systems. Icfain J. Manage. Res., 48–64 (2005)

18. Teacy, W.T.L., Patel, J., Jennings, N.R., Luck, M.: Coping with inaccurate reputation sources: experimental analysis of a probabilistic trust model. In: Proceedings of Fourth International Autonomous Agents and Multiagent Systems (AAMAS), pp. 997–1004. (2005)
19. Sabater, J., Sierra, C.: Regret: A reputation model for gregarious societies. In: Proceedings of the Fifth International Conference on Autonomous Agents Workshop on Deception, Fraud and Trust in Agent Societies, pp. 61–69. (2001)
20. Kamvar, S., Schlosser, M., Garcia-Molina, H.: The eigentrust algorithm for reputation management in p2p networks. In: WWW '03: Proceedings of the 12th International Conference on World Wide Web, pp. 640–651. ACM Press (2003)
21. Jiang, T., Baras, J.: Trust evaluation in anarchy: a case study on autonomous networks. In: INFOCOM, IEEE (2006)
22. Sun, Y., Han, Z., Yu, W., Liu, K.: A trust evaluation framework in distributed networks: vulnerability analysis and defense against attacks. In: INFOCOM, IEEE (2006)
23. Xiong, L., Liu, L.: Peertrust: supporting reputation-based trust for peer-to-peer electronic communities. IEEE Trans. Knowl. Data Eng. **16**(7), 843–857 (2004)
24. Mekouar, L., Iraqi, Y., Boutaba, R.: Reputation-based trust management in peer-to-peer systems: taxonomy and anatomy. In: Handbook of Peer-to-Peer Networking, pp. 689–732. (2010)
25. Rahbar, A., Yang, O.: Powertrust: a robust and scalable reputation system for trusted peer-to-peer computing. IEEE Trans. Parallel Distrib. Syst. **18**(4), 460–473 (2007)
26. Srivatsa, M., Xiong, L., Liu, L.: TrustGuard: countering vulnerabilities in reputation management for decentralized overlay networks. In: Proceedings of the 14th International Conference on World Wide Web, pp. 422–431. ACM New York, NY, USA (2005)
27. Zhang, Y., Fang, Y.: A fine-grained reputation system for reliable service selection in peer-to-peer networks. IEEE Trans. Parallel Distrib. Syst., 1134–1145 (2007)

Author Biographies

Carol Fung is a Ph.D. student in University of Waterloo (Canada). Her research topic is collaborative Intrusion Detection networks, which includes trust management, collaborative decision, resource management, and incentive design of such a system. She is also interested in the security problems in wireless networks and social networks.

Jie Zhang joined Nanyang Technological University, Singapore as an assistant professor, working on trust modeling and incentive mechanism design. He got Ph.D. in the School of Computer Science at the University of Waterloo, and was the recipient of the Alumni Gold Medal, awarded to honour the top Ph.D. graduate.

Issam Aib is currently a research fellow in the School of Computer Science at the University of Waterloo (Canada). He received Ph.D. from the University of Pierre et Marie Curie. His research interests include policy-based management, security policies, intrusion detection systems, and management of WLANs.

Raouf Boutaba is a Professor of Computer Science and a Cheriton Faculty Fellow at the University of Waterloo. His research interests include network, resource and service management. He is the founding Editor-in-Chief of IEEE Transactions on Network and Service Management. He has received several recognitions (i.e. the Premiers research excellence award).